

# Python: module vcs.yxvsx

## vcs.yxvsx

[index](#)

# Yxvsx (GYx) module

## Modules

[vcs.Canvas](#)  
[Numeric](#)

[vcs.VCS\\_validation\\_functions](#) [cdms](#)  
[vcs.vcs](#) [cdtime](#)

[vcs.queries](#)  
[string](#)

## Classes

[builtin.object](#)  
[GYx](#)

class **GYx**([builtin.object](#))

Class: [GYx](#) # Yxvsx

Description of [GYx](#) Class:

The Yxvsx graphics method displays a line plot from a 1D data array (a plot of  $Y(x)$ , where  $y$  represents the 1D coordinate values). The class shows how to change line and marker attributes for the Yxvsx graphics.

This class is used to define an Yxvsx table entry used in VCS, or used to change some or all of the Yxvsx attributes in an existing entry.

Other Useful Functions:

a=vcs.init()	# Constructor
a.show('yxvsx')	# Show predefined Yxvsx graph
a.show('line')	# Show predefined VCS line ob
a.show('marker')	# Show predefined VCS marker
a.setcolormap("AMIP")	# Change the VCS color map
a.yxvsx(s, x, 'default')	# Plot data 's' with Yxvsx and 'default' template
a.update()	# Updates the VCS Canvas at
a.mode=1, or 0	If 1, then automatic update 0, then use update function

Example of Use:

a=vcs.init()

To Create a new instance of Yxvsx use:

yxx=a.createyvsvy('new','quick') # Copies content of 'quick'

```

yxx=a.createxyvsv('new')                                # Copies content of 'default'

To Modify an existing Yxvsx use:
yxx=a.getxyvsv('AMIP_psl')

yxx.list()                                              # Will list all the Yxvsx at-
yxx.projection='linear'                                 # Can only be 'linear'
lon30={-180:'180W',-150:'150W',0:'Eq'}
yxx.xticlabels1=lon30
yxx.xticlabels2=lon30
yxx.xticlabels(lon30, lon30)                          # Will set them both
yxx.xmtics1=''
yxx.xmtics2=''
yxx.xmtics(lon30, lon30)                            # Will set them both
yxx.yticlabels1=lat10
yxx.yticlabels2=lat10
yxx.yticlabels(lat10, lat10)                          # Will set them both
yxx.ymtics1=''
yxx.ymtics2=''
yxx.ymtics(lat10, lat10)                            # Will set them both
yxx.datawc_y1=-90.0
yxx.datawc_y2=90.0
yxx.datawc_x1=-180.0
yxx.datawc_x2=180.0
yxx.datawc(-90, 90, -180, 180)                      # Will set them all
yxx.xaxisconvert='linear'

Specify the Yxvsx line type:
yxx.line=0                                            # same as yxx.line = 'solid'
yxx.line=1                                            # same as yxx.line = 'dash'
yxx.line=2                                            # same as yxx.line = 'dot'
yxx.line=3                                            # same as yxx.line = 'dash-dot'
yxx.line=4                                            # same as yxx.line = 'long-dash'

Specify the Yxvsx line color:
yxx.linecolor=16      # color range: 16 to 230, default color is black
yxx.linewidth=1       # width range: 1 to 100, default color is 1

Specify the Yxvsx marker type:
yxx.marker=1                                         # Same as yxx.marker='dot'
yxx.marker=2                                         # Same as yxx.marker='plus'
yxx.marker=3                                         # Same as yxx.marker='star'
yxx.marker=4                                         # Same as yxx.marker='circle'
yxx.marker=5                                         # Same as yxx.marker='cross'
yxx.marker=6                                         # Same as yxx.marker='diamond'
yxx.marker=7                                         # Same as yxx.marker='triangle'
yxx.marker=8                                         # Same as yxx.marker='triangledown'
yxx.marker=9                                         # Same as yxx.marker='triangleright'
yxx.marker=10                                         # Same as yxx.marker='triangleftarrow'
yxx.marker=11                                         # Same as yxx.marker='square'
yxx.marker=12                                         # Same as yxx.marker='diamond'
yxx.marker=13                                         # Same as yxx.marker='triangleright'

```

```

yxx.marker=14          # Same as yxx.marker='triangle'
yxx.marker=15          # Same as yxx.marker='triangle'
yxx.marker=16          # Same as yxx.marker='triangle'
yxx.marker=17          # Same as yxx.marker='square'
yxx.marker=None        # Draw no markers

```

There are four possibilities for setting the marker color index

```

yxx.markercolors=22      # Same as below
yxx.markercolors=(22)    # Same as below
yxx.markercolors=([22])  # Will set the markers to a
                        # color index
yxx.markercolors=None   # Color index defaults to Black

```

To set the Yxvsx Marker size:

```

yxx.markersize=5
yxx.markersize=55
yxx.markersize=100
yxx.markersize=300
yxx.markersize=None

```

Methods defined here:

```

__init__(self, parent, GYx_name=None, GYx_name_src='default', createGYx=0)

datawc(self, dsp1=1e+20, dsp2=1e+20, dsp3=1e+20, dsp4=1e+20)

list(self)

rename = renameGYx(self, old_name, new_name)
    ######
    #
    # Function:      renameGYx
    #
    # Description of Function:
    #     Private function that renames the name of an existing
    #     graphics method.
    #
    #
    # Example of Use:
    #     renameGYx(old_name, new_name)
    #             where: old_name is the current name of Yxvsx or
    #                   new_name is the new name for the Yxvsx
    #
    #####

```

**script(self, script\_filename=None, mode=None)**

Function: script # Calls \_vcs.s

Description of Function:  
Saves out a boxfill graphics method in Python or VCS source  
designated file.

Example of Use:

`script(scriptfile_name, mode)`

where: `scriptfile_name` is the output name of the  
mode is either "w" for replace or "a" for append

Note: If the filename has a ".py" at the end it will produce a Python script. If the filename has a ".scr" extension it will produce a VCS script. If neither extension is present the default a Python script will be produced.

```
a=vcs.init()  
Yx=a.createboxfill('temp')  
Yx.script('filename.py')           # Append to a Python file  
Yx.script('filename.scr')         # Append to a VCS file "filename.scr"  
Yx.script('filename', 'w')
```

**xmtics**(self, xmt1=", xmt2="")

**xticlabels**(self, xtl1=", xtl2="")

**ymtics**(self, ymt1=", ymt2="")

**yticlabels**(self, ytl1=", ytl2="")

---

Properties defined here:

**datawc\_calendar**

```
get">get = _getcalendar(self)  
set">set = _setcalendar(self, value)
```

**datawc\_timeunits**

```
get">get = _gettimeunits(self)  
set">set = _settimeunits(self, value)
```

**datawc\_x1**

```
get">get = _getdatawc_x1(self)  
set">set = _setdatawc_x1(self, value)
```

**datawc\_x2**

```
get">get = _getdatawc_x2(self)  
set">set = _setdatawc_x2(self, value)
```

**datawc\_y1**

```
get">get = _getdatawc_y1(self)  
set">set = _setdatawc_y1(self, value)
```

**datawc\_y2**

```
get">get = _getdatawc_y2(self)  
set">set = _setdatawc_y2(self, value)
```

**line**

```
get">get = _getline(self)
set">set = _setline(self, value)

linecolor
get">get = _getlinecolor(self)
set">set = _setlinecolor(self, value)

linewidth
get">get = _getlinewidth(self)
set">set = _setlinewidth(self, value)

marker
get">get = _getmarker(self)
set">set = _setmarker(self, value)

markercolor
get">get = _getmarkercolor(self)
set">set = _setmarkercolor(self, value)

markersize
get">get = _getmarkersize(self)
set">set = _setmarkersize(self, value)

name
get">get = _getname(self)
set">set = _setname(self, value)

projection
get">get = _getprojection(self)
set">set = _setprojection(self, value)

xaxisconvert
get">get = _getxaxisconvert(self)
set">set = _setxaxisconvert(self, value)

xmtics1
get">get = _getxmtics1(self)
set">set = _setxmtics1(self, value)

xmtics2
get">get = _getxmtics2(self)
set">set = _setxmtics2(self, value)

xticlabels1
get">get = _getxticlabels1(self)
set">set = _setxticlabels1(self, value)

xticlabels2
get">get = _getxticlabels2(self)
set">set = _setxticlabels2(self, value)

yntics1
```

```

    get">get = _getymtics1(self)
    set">set = _setymtics1(self, value)

ymtics2
    get">get = _getymtics2(self)
    set">set = _setymtics2(self, value)

yticlabels1
    get">get = _getyticlabels1(self)
    set">set = _setyticlabels1(self, value)

yticlabels2
    get">get = _getyticlabels2(self)
    set">set = _setyticlabels2(self, value)

```

---

Data and other attributes defined here:

```

__slots__ = ['setmember', 'parent', 'name', 'g_name', 'xaxisconvert', 'linecolor', 'line', 'linewidth', 'ma
'markercolor', 'projection', 'xticlabels1', 'xticlabels2', 'yticlabels1', 'yticlabels2', 'xmtics1', 'xmtics2', '']

g_name = <member 'g_name' of 'GYx' objects>

parent = <member 'parent' of 'GYx' objects>

setmember = <member 'setmember' of 'GYx' objects>

```

## Functions

```

getGYxmember(self, member)
#####
#
# Function:      getGYxmember
#
# Description of Function:
#     Private function that retrieves the Yxvsx members from the
#     structure and passes it back to Python.
#
#
# Example of Use:
#     return_value =
#         getGYxmember(self, name)
#             where: self is the class (e.g., GYx)
#                     name is the name of the member that is being
#
#####
getmember = getGYxmember(self, member)
#####
#

```

```

# Function:      getGYxmember
#
# Description of Function:
#     Private function that retrieves the Yxvsx members from the
#     structure and passes it back to Python.
#
#
# Example of Use:
#     return_value =
#     getGYxmember(self, name)
#             where: self is the class (e.g., GYx)
#                     name is the name of the member that is being
#
#####
renameGYx(self, old_name, new_name)
#####
#
# Function:      renameGYx
#
# Description of Function:
#     Private function that renames the name of an existing Yxvsx
#     graphics method.
#
#
# Example of Use:
#     renameGYx(old_name, new_name)
#             where: old_name is the current name of Yxvsx graphi
#                     new_name is the new name for the Yxvsx graphi
#
#####
setGYxmember(self, member, value)
#####
#
# Function:      setGYxmember
#
# Description of Function:
#     Private function to update the VCS canvas plot. If the can
#     set to 0, then this function does nothing.
#
#
# Example of Use:
#     setGYxmember(self, name, value)
#             where: self is the class (e.g., GYx)
#                     name is the name of the member that is being
#                     value is the new value of the member (or att
#
#####
setmember = setGYxmember(self, member, value)

```

```
#####
#
# Function:      setGYxmember
#
# Description of Function:
#     Private function to update the VCS canvas plot. If the can
#     set to 0, then this function does nothing.
#
#
# Example of Use:
#     setGYxmember(self,name,value)
#             where: self is the class (e.g., GYx)
#                     name is the name of the member that is being
#                     value is the new value of the member (or att
#
#####
#
```

## Data

*StringTypes* = (<type 'str'>, <type 'unicode'>)